REMARKS

CLAIM AMENDMENTS

CLAIMS 19, 20, 44, AND 45

An incorrect grammatical form for "enable" was used in the original claims. The grammatical form is corrected by the amendments.

35 U.S.C. § 102 CLAIM REJECTIONS

"A claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described, in a single prior art reference." *Verdegaal Bros. v. UnionOil Co. of California,* 2 USPQ2d 1051, 1053 (Fed. Cir. 1987). (Cited in MPEP § 2131.)

"The identical invention must be shown in as complete detail as is contained in the . . . claim." *Richardson v. Suzuki Motor Co.*, 9 USPQ2d 1913, 1920 (Fed. Cir. 1989). (Cited in MPEP § 2131.)

CLAIM 1

Claim 1 reads as follows:

1. *A method **[1] for repetitively executing a plurality of software packages at one or more rates**, utilizing a common set of computational resources, the method comprising the steps:*

[2] generating a sequence of time intervals for each of the plurality of software packages, the time intervals belonging to one software package not overlapping the time intervals belonging to any other of the plurality of software packages;

[3] executing a plurality of software packages, each software package being executed during the time intervals of its sequence of time intervals.

The limitations shown in boldface are not disclosed by Maitra.

Limitation [1]

Maitra does not disclose repetitively executing a plurality of software packages at one or more rates. The concept of a software package "execution rate" is not to be found in Maitra. In fact, Maitra's disclosure indicates that there is no such thing as an "execution rate" for an application executed by the Maitra invention. Specifically, Maitra discloses how applications are executed in the following words:

> "Typically the task scheduling unit 110 assigns each application a small unit of time, called a time quantum or time slice, when it is scheduled to run on the microprocessor 140. . . . **If the application has blocked or finished before the quantum has elapsed, the microprocessor switches to the next application.**" Maitra, col. 4, lines 32-44 (emphasis added).

To illustrate, consider three applications A, B, and C which are assigned by the task scheduling unit 110 to run according to a round-robin scheduling scheme. Maitra, col. 4, lines 32-42. Applications A, B, and C are assigned time slices equivalent respectively to three letter time periods, five letter time periods, and two letter time periods as indicated in the following sequence.

. . . |AAABBBBBCC|AAABBBBBCC|AAABBBBBCC|AAABBBBBCC| . . .

The vertical separators denote the end of one round-robin cycle and the beginning of the next.

The portion of a "time quantum" utilized by an application depends on the task being performed by the application. For example, in a spread-sheet application, the execution of a "page compute" takes more time than the execution of "record data entries". Maitra, col. 7, lines 38-44. As a result, the time interval between repeat executions of a given application is essentially indeterminate and an execution rate of the application cannot be specified.

To illustrate, let us assume that application B requires three letter time periods to finish during the first round-robin cycle, one letter time period to finish during the second round-robin cycle, five letter time periods to finish during the third round-robin cycle, and two letter time periods to finish during the fourth round-robin cycle. Maitra states that under such circumstances "[if] the application has blocked or finished before the quantum has elapsed, the microprocessor switches to the next application." Maitra, col. 4, lines 42-44. Thus, in this situation the above sequence becomes:

. . . |AAABBBCC|AAABCC|AAABBBBBCC|AAABBCC| . . .

Note how the cycle period continually changes. As a result, one cannot say that there is a particular execution rate for any of the applications as called for by limitation [1].

It is sometimes argued that a limitation appearing in the preamble of a claim such as limitation [1] is merely a statement of the intended use of the device. However, the Manual of Patent Examining Procedure emphasizes:

> **"The claim preamble must be read in the context of the entire claim. The determination of whether preamble recitations are structural limitations or mere statements of purpose or use 'can be resolved only on review of the entirety of the [record] to gain an understanding of what the inventors actually invented and intended to encompass by the claim.' *Corning Glass Works v. Sumitomo Elec. U.S.A., Inc.*, 868 F.2d 1251, 1257, 9 USPQ2d 1962, 1966 (Fed. Cir. 1989)."** MPEP § 2111.02.

For example, the "for use" clause in "a bicycle for use in getting from one place to another" is merely a statement of use and does not imply any structural limitations on the bicycle. On the other hand, the "for use" clause in "a cell phone for use in the Verizon Wireless network" is not merely a statement of use but also implies certain structural limitations on the cell phone so that it has the capability of being used in the Verizon Wireless network.

In the case of applicants' "*a method [1] for repetitively executing a plurality of software packages at one or more rates*", the "for use" clause, like the cell phone example, is not merely a

statement of use but also implies limitations to the functions performed in the steps of the method. In such cases, the Manual of Patent Examining Procedure states:

> **"Any terminology in the preamble that limits the structure of the claimed invention must be treated as a claim limitation.** ***See, e.g., Corning Glass Works v. Sumitomo Elec. U.S.A., Inc.*****, 868 F.2d 1251, 1257, 9 USPQ2d 1962, 1966 (Fed. Cir. 1989);** ***Pac-Tec Inc. v. Amerace Corp.*****, 903 F.2d 796, 801, 14 USPQ2d 1871, 1876 (Fed. Cir. 1990).** ***See also In re Stencel,*** **828 F.2d 751, 4 USPQ2d 1071 (Fed. Cir. 1987)"** MPEP § 2111.02.

One might ask whether applicants' "*a method [1] for repetitively executing a plurality of software packages at one or more rates*" asserts limitations that are not already fully and intrinsically set forth by the limitations in the body of the claim:

> **"If the body of a claim fully and intrinsically sets forth all of the limitations of the claimed invention, and the preamble merely states, for example, the purpose or intended use of the invention, rather than any distinct definition of any of the claimed invention's limitations, then the preamble is not considered a limitation and is of no significance to claim construction.** ***Pitney Bowes, Inc. v. Hewlett-Packagd Co.*****, 182 F.3d 1298, 1305, 51 USPQ2d 1161, 1165 (Fed. Cir. 1999)."** MPEP § 2111.02.

Neither of the body elements of the claim at issue disclose either separately or in combination "*a method [1] for repetitively executing a plurality of software packages at one or more rates*", and thus, these words represent an additional structural limitation which must be treated as a legitimate claim limitation and not merely "a statement of purpose or use."

Limitation [2]

(2) Maitra does not disclose ***generating a sequence of non-overlapping time intervals for each of the plurality of software packages.***

Maitra discloses a system where "[a]t the end of the time slice, the process is suspended and the next scheduled process is readied for execution." Maitra, col. 4, lines 40-42. In the round-robin scheduling approach, a list of applications that need to be run is maintained and the

task scheduling unit 110 runs each application in the order that it appears on the list. Maitra, col. 4, lines 37-40. Nothing is disclosed in Maitra concerning the "generating a sequence of non-overlapping time intervals for each of the plurality of software packages". If such a sequence had been generated, Maitra would necessarily base his selection of an application to be performed by first identifying which sequence of non-overlapping time intervals contained the upcoming time interval and then determining which application was associated with that sequence.

The examiner suggests that this limitation is disclosed by Maitra at col. 6, lines 34-39 which describes a situation where two applications execute every 10 milliseconds. But there is nothing in this passage that suggests "generating a sequence of non-overlapping time intervals for each of the plurality of software packages." This limitation requires that each of the plurality of software packages must be associated with an identifiable sequence of time intervals. A disclosure of two applications executing every 10 milliseconds is not a disclosure of generating (i.e. bringing into existence) a plurality of sequences of non-overlapping time intervals (one for each application to be executed) and associating each of the sequences with an application to be executed.

Limitation [3]

(3) Maitra does not disclose ***executing a plurality of software packages, each software package being executed during the time intervals of its sequence of time intervals.***

To disclose this limitation, Maitra's apparatus would have to (1) identify the beginning of the next time period, (2) identify the sequence of time periods of which the next time period is a member, (3) identify the application associated with the identified sequence of time periods, and (4) execute the identified application beginning with the start of the next time period. Rather than performing this process, Maitra's task scheduling unit 110 simply waits for the presently-

executing application to quit or finish processing and causes the next application on the list of applications to be executed. Maitra, col. 4, lines 37-44. Maitra's approach to selecting an application for execution has nothing to do with selecting an application based on its association with the sequence of time periods of which the upcoming time period is a member.

The examiner suggests that limitation [3] is disclosed by the passage in Maitra at col. 6, lines 40-55: "At the end of the time slice, the process is suspended and the next scheduled process is readied for execution. If the application has blocked or finished before the quantum has elapsed, the microprocessor switches to the next application." But this passage says nothing about identifying the application to be executed as the application associated with the sequence of time periods of which the upcoming time period is a member. And Maitra's apparatus does not generate the specified sequences of time periods to provide a timing basis for switching from a presently-executing application to the next. Instead, Maitra's apparatus simply begins execution of another application whenever the presently-executing application finishes.

Maitra does not disclose any of the boldface limitations of claim 1 and consequently did not anticipate applicants' claim-1 invention.

CLAIM 25

Claim 25 includes all of the limitations of claim 1 and consequently was also not anticipated by Maitra.

CLAIM 26

Claim 26 is a claim for apparatus that includes all of the limitations of method claim 1. Consequently, as in the case of applicants' claim-1 invention, applicants' claim-26 invention also was not anticipated by Maitra.

CLAIM 27

Claim 27 reads as follows:

*27. The apparatus of claim 26 wherein **[1] the plurality of software packages executed by the "executing" means includes only valid software packages**, the apparatus further comprising:*

[2] a means for utilizing one or more tests to identify the software packages that are valid.

(4) Maitra does not require that ***[1] the plurality of software packages executed by the "executing" means includes only valid software packages.***

(5) Maitra does not disclose ***[2] a means for utilizing one or more tests to identify the software packages that are valid.***

The passages from Maitra cited by the examiner in connection with claim 1 say nothing about "valid" software packages or tests for identifying software packages that are "valid".

Maitra does not disclose the limitations in boldface of claim 27, and consequently Maitra did not anticipate applicants' claim-27 invention.

CLAIM 2

Claim 2 reads as follows:

*2. The method of claim 1 wherein **[1] the plurality of software packages of the "executing" step includes only valid software packages, the method further comprising the step:***

[2] utilizing one or more tests to identify the software packages that are valid.

Maitra does not require that ***[1] the plurality of software packages executed by the "executing" step includes only valid software packages.***

Maitra does not disclose ***[2] utilizing one or more tests to identify the software packages that are valid.***

The passage from Maitra cited by the examiner (col. 6, lines 56-67) has to do with establishing the proper clock speed for running an application (see col. 7, lines 1-3). The disclosure has nothing to do with determining the "validity" of applications and executing only "valid" applications. In the Maitra invention, every application can be executed simply by adjusting the clock speed of the computer system (see Maitra, col. 7, lines 1-3).

Maitra does not disclose the limitations in boldface of claim 2, and consequently Maitra did not anticipate applicants' claim-2 invention.

CLAIMS 6 AND 31

Claim 6 reads as follows:

6. *The method of claim 2* ***wherein [1] a software package is assigned its own dedicated memory region, [2] the software package's dedicated memory region including a stack memory region and/or a heap memory region, [3] one of the tests for validity being whether the stack memory range and/or the heap memory range assigned during the execution of the software package's initialization procedure and the various associated entry points lies within the software package's dedicated memory region.***

Apparatus claim 31 also contains limitations [1], [2], and [3].

Maitra does not disclose any of the limitations shown above in boldface. The examiner cites Maitra, col. 7, line 66, through col. 8, line 20, as disclosing all of the limitations of claims 6 and 31, but it does not.

Maitra discloses a single memory for storing "a first plurality of processor executable instructions, a second plurality of processor executable instructions and a third plurality of processor executable instructions" (col. 8, lines 7-10) but says nothing about these three sets of executable instructions being stored in dedicated regions of memory.

Nor does Maitra say anything about "a software package's dedicated memory region including a stack memory region and/or a heap memory region."

Nor is the test of validity "whether the stack memory range and/or the heap memory range assigned during the execution of the software package's initialization procedure and the various associated entry points lies within the software package's dedicated memory region" disclosed in the Maitra passage.

Maitra does not disclose the limitations in boldface and consequently Maitra did not anticipate applicants' claim-6 and claim-31 inventions.

CLAIMS 8 AND 33

Claim 8 reads as follows:

8. The method of claim 1 wherein a software package is assigned its own dedicated memory region.

Apparatus claim 33 contains the same limitation as claim 8.

Maitra does not disclose the limitation of claims 8 and 33. The examiner cites Maitra, col. 7, line 66, through col. 8, line 20, as disclosing the limitation of claims 8 and 33, but it does not.

Maitra discloses a single memory for storing "a first plurality of processor executable instructions, a second plurality of processor executable instructions and a third plurality of processor executable instructions" (col. 8, lines 7-10) but says nothing about these three sets of executable instructions being stored in dedicated regions of memory.

Maitra does not disclose the limitation of claims 8 and 33 and consequently Maitra did not anticipate applicants' claim-8 and claim-33 inventions.

CLAIMS 9 AND 34

Claim 9 reads as follows:

9. The method of claim 8 wherein the software package's dedicated memory region includes a stack memory region, a software package's stack residing in the software package's stack memory region.

Apparatus claim 34 contains the same limitation as claim 9.

Maitra does not disclose the limitation of claims 9 and 34. The examiner cites Maitra, col. 7, line 66, through col. 8, line 20, as disclosing the limitation of claims 8 and 33, but it does not.

Maitra says nothing about the software package's dedicated memory region including a stack memory region nor does it say anything about a software package's stack residing in the software package's stack memory region.

Maitra does not disclose the limitations of claims 9 and 34 and consequently Maitra did not anticipate applicants' claim-9 and claim-34 inventions.

CLAIMS 7 AND 32

Claim 7 reads as follows:

7. *The method of claim 6 wherein one of the tests is whether the stack memory range and/or the heap memory range and the various associated entry points are returned within a predetermined time.*

Apparatus claim 32 contains the same limitations as claim 7.

Maitra does not disclose the limitations of claims 7 and 32. The examiner cites Maitra, col. 3, line 66, through col. 8, line 20, as disclosing the limitation of claims 7 and 32, but it does not.

The cited passage of Maitra discloses how the operating frequency of the microprocessor is adjusted to meet the computing requirements of the processes. Maitra says nothing about performing tests to determine the validity of the applications to be executed, and more specifically, says nothing about a test of validity being "whether the stack memory range and/or the heap memory range and the various associated entry points are returned within a predetermined time."

Maitra does not disclose the limitations of claims 7 and 32 and consequently Maitra did not anticipate applicants' claim-7 and claim-32 inventions.

CLAIMS 21 AND 46

Claim 21 reads as follows:

21. The method of claim 1 wherein an executive software package enforces the discipline that each software package executes only during the time intervals of its sequence of time intervals, the executive software package determining when the execution of a software package extends into a time interval belonging to the sequence of time intervals assigned to another software package and performs a remedial action.

Apparatus claim 46 contains the same limitations as claim 21.

Maitra does not disclose the limitations of claims 21 and 46. The examiner cites Maitra, col. 4, lines 22-35, as disclosing the limitation of claims 21 and 46, but it does not.

The cited passage of Maitra discloses how the "[t]ask scheduling unit 110 is coupled to microprocessor 140 and schedules the applications which are run by the microprocessor 140." As Maitra points out, the execution of an application is suspended at the end of its assigned "time slice" (col. 4, lines 40-42). The Maitra invention does not allow the execution of an application to extend into another application's assigned "time slice". This is not a disclosure of the limitations of claims 21 and 46 which envisions the possibility of the execution of a software package extending into another software package's assigned time interval and requiring a remedial action.

Maitra does not disclose the limitations of claims 21 and 46 and consequently Maitra did not anticipate applicants' claim-21 and claim-46 inventions.

CLAIMS 22 AND 47

Claim 22 reads as follows:

22. The method of claim 1 wherein the presence of those software packages that are present is detected.

Apparatus claim 47 contains the same limitation as claim 22.

Maitra does not disclose the limitations of claims 22 and 47. The examiner cites Maitra, col. 4, lines 22-35, as disclosing the limitation of claims 21 and 46, but it does not.

The cited passage of Maitra discloses that the "[t]ask scheduling unit 110 is coupled to microprocessor 140 and schedules the applications which are run by the microprocessor 140, the order in which the applications are run, and the amount of CPU time each application receives." Maitra discloses that scheduling unit 110 schedules THE APPLICATIONS WHICH ARE RUN BY THE MICROPROCESSOR 140. This passage says nothing about DETECTING THE PRESENCE OF APPLICATIONS TO BE RUN.

Maitre discloses that "the task scheduler 110 maintains a list of applications to be run" (col. 4, lines 37-38) but maintaining a list of applications to be run is not the same as DETECTING THE PRESENCE OF APPLICATIONS TO BE RUN.

Maitra does not disclose the limitation of claims 22 and 47 and consequently Maitra did not anticipate applicants' claim-22 and claim-47 inventions.

35 U.S.C. § 103 CLAIM REJECTIONS

Three basic criteria must be satisfied to establish *prima facie* obviousness:

"The legal concept of *prima facie* obviousness is a procedural tool of examination which applies broadly to all arts. . . . The examiner bears the initial burden of factually supporting any *prima facie* conclusion of obviousness. If the examiner does not produce a *prima facie* case, the applicant is under no obligation to submit evidence of nonobviousness. . . . To establish a *prima facie* case of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. The teaching or suggestion to make the claimed combination and the reasonable expectation of success must both be found in the prior art, and not based

on applicant's disclosure. *In re Vaeck*, 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991)." MPEP § 2142.

CLAIMS 3 AND 28

Claim 3 reads as follows:

3. *The method of claim 2 wherein one of the tests for validity is a one's complement checksum test of a software package's program memory.*

Apparatus claim 28 contains the same limitation as claim 3.

Neither of the cited references discloses the limitation of claim 3. The passage in Jablon et al. cited by the examiner (col. 5, lines 4-18) describes the use of checksums to validate "extension programs" to the BIOS program which resides in the read-only memory of a PC. Jablon et al. says nothing about applying checksums in general to a software package's program memory. Nor does Jablon et al. say anything about the use of one's complement checksums in testing the validity of a software package.

Even if Jablon et al. did disclose applicants' limitation, there is no motivation to be found in either Maitra of Jablon et al. for a person skilled in the art including such a limitation in Maitra's invention. Certainly Maitra expresses no concern as to the need for testing the validity of the applications that his invention is intended to execute. And Jablon et al. actually teaches away from the use of checksums by stating that (col. 5, lines 15-18) the "storage protection method [of U.S. Pat. No. 5,022,077]shares the architectural weakness of most software-controlled protection schemes on the PC."

Thus, Maitra and Jablon et al. in combination do not meet two of the basic criteria for establishing *prima facie* obviousness of claims 3 and 28. Neither of the references discloses

applicants' one's complement test for software validity and there is no motivation for providing such a test in the Maitra invention.

CLAIMS 4 AND 29

Claim 4 reads as follows:

4. *The method of claim 2 wherein a software package is assigned its own dedicated memory region, one of the tests for validity being whether the address returned for the software package's initialization procedure lies within its dedicated memory region.*

Apparatus claim 29 contains the same limitation as claim 4.

Neither of the cited references discloses the limitation of claim 4. The passage in Jablon et al. cited by the examiner (col. 6, lines 27-37) points out that methods for partitioning memory "generally allow trusted software to both enable and disable the protection mechanism for a given region of memory" and that the memory protection feature in the Jablon et al. invention only allows software control from unprotected to protected mode. Nothing is said about "one of the tests for validity being whether the address returned for the software package's initialization procedure lies within its dedicated memory region."

Since neither Maitra nor Jablon et al. disclose the limitations of claims 4 and 29, there is obviously no motivation for a person skilled in the art for incorporating such limitations in the Maitra invention.

Thus, Maitra and Jablon et al. in combination do not meet two of the basic criteria for establishing *prima facie* obviousness of claims 4 and 29. Neither of the references discloses applicants' address-returned test for software validity and there is no motivation for providing such a test in the Maitra invention.

CLAIMS 5 AND 30

Claim 5 reads as follows:

5. The method of claim 4 wherein one of the tests is whether the address is returned within a predetermined time.

Apparatus claim 30 contains the same limitation as claim 5.

Neither of the cited references discloses the limitations of claim 4s and 29 (see discussion above). The references also do not disclose the additional limitation of claims 5 and 30 which further limits the limitations of claims 4 and 29.

Thus, Maitra and Jablon et al. in combination do not meet two of the basic criteria for establishing *prima facie* obviousness of claims 5 and 30. Neither of the references discloses applicants' address-returned within a predetermined time test for software validity and there is no motivation for providing such a test in the Maitra invention.

CLAIMS 18 AND 43

Claim 18 reads as follows:

18. The method of claim 1 wherein safety-critical software is placed in one or more separate partitions thereby isolating the safety-critical software from non-safety-critical software.

Apparatus claim 43 contains the same limitation as claim 18.

Neither of the cited references discloses the limitation of claim 18. The passage in Jablon et al. cited by the examiner (col. 6, lines 27-38) points out that methods for partitioning memory "generally allow trusted software to both enable and disable the protection mechanism for a given

region of memory" and that the memory protection feature in the Jablon et al. invention only allows software control from unprotected to protected mode. Nothing is said about "safety-critical software . . . [being] placed in one or more separate partitions thereby isolating the safety-critical software from non-safety-critical software."

Since neither Maitra nor Jablon et al. disclose the limitations of claims 18 and 43, there is obviously no motivation for a person skilled in the art for incorporating such limitations in the Maitra invention.

Thus, Maitra and Jablon et al. in combination do not meet two of the basic criteria for establishing *prima facie* obviousness of claims 18 and 43. Neither of the references discloses applicants' safety-critical software isolation limitation and there is no motivation for providing such a test in the Maitra invention.

CLAIMS 19 AND 44

Claim 19 reads as follows:

19. The method of claim 1 wherein each of the plurality of software packages is assigned its own memory block, a software package being enableable to read data only from zero or more memory blocks associated with other software packages, the zero or more memory blocks readable by a software package being either predetermined or determined during execution of the software packages in accordance with a set of one or more rules.

Apparatus claim 44 contains the same limitations as claim 19.

Neither of the cited references discloses the limitation of claim 19. The passage in Jablon et al. cited by the examiner (col. 8, lines 16-33) discloses a system wherein "during system initialization trusted software closes the latch to protect the memory, and thus prevent all

subsequently run programs from reading and/or writing the security-relevant data during normal operation." This is not a disclosure of "a software package being enableable to read data only from zero or more memory blocks associated with other software packages, the zero or more memory blocks readable by a software package being either predetermined or determined during execution of the software packages in accordance with a set of one or more rules."

Since neither Maitra nor Jablon et al. disclose the limitations of claims 19 and 44, there is obviously no motivation for a person skilled in the art for incorporating such limitations in the Maitra invention.

Thus, Maitra and Jablon et al. in combination do not meet two of the basic criteria for establishing *prima facie* obviousness of claims 19 and 44. Neither of the references discloses applicants' limitations and there is no motivation for providing such limitations in the Maitra invention.

CLAIMS 20 AND 45

Claim 20 reads as follows:

20. The method of claim 1 wherein each of the plurality of software packages is assigned its own memory block, a software package being enableable to write data only to zero or more memory blocks associated with other software packages, the zero or more memory blocks writeable by a software package being either predetermined or determined during execution of the software packages in accordance with a set of one or more rules.

Apparatus claim 45 contains the same limitations as claim 20.

Neither of the cited references discloses the limitation of claim 20. The passage in Jablon et al. cited by the examiner (col. 8, lines 16-33) discloses a system wherein "during system

initialization trusted software closes the latch to protect the memory, and thus prevent all subsequently run programs from reading and/or writing the security-relevant data during normal operation." This is not a disclosure of "a software package being enableable to write data only from zero or more memory blocks associated with other software packages, the zero or more memory blocks writeable by a software package being either predetermined or determined during execution of the software packages in accordance with a set of one or more rules."

Since neither Maitra nor Jablon et al. disclose the limitations of claims 20 and 45, there is obviously no motivation for a person skilled in the art for incorporating such limitations in the Maitra invention.

Thus, Maitra and Jablon et al. in combination do not meet two of the basic criteria for establishing *prima facie* obviousness of claims 20 and 45. Neither of the references discloses applicants' limitations and there is no motivation for providing such limitations in the Maitra invention.

CLAIMS 10 AND 35

Claim 10 reads as follows:

10. The method of claim 1 wherein a software package includes background tasks as well as foreground tasks, the background tasks being performed after the foreground tasks have been completed.

Apparatus claim 35 contains the same limitations as claim 10.

Neither of the cited references discloses the limitation of claim 10. The passage in Reznak cited by the examiner (col. 1, lines 25-32) discloses (1) cooperative multitasking operating systems wherein background tasks are granted processing time only during idle periods

of the foreground tasks and (2)time-slice multitasking operating systems wherein processing time is allocated to each task, be it foreground or background, in round-robin fashion or based upon task priority. Neither the cooperative multitasking operating system nor the time-slice multitasking operating system discloses applicants' limitation "the background tasks being performed after the foreground tasks have been completed." The cooperative multitasking operating system is disclosed as performing background tasks during the idle periods between foreground tasks. The time-slice multitasking operating system is disclosed as performing tasks which may be either foreground or background in a round-robin fashion or based upon task priority.

Since neither Maitra nor Reznak disclose the limitations of claims 10 and 35, there is obviously no motivation for a person skilled in the art for incorporating such limitations in the Maitra invention.

Thus, Maitra and Reznak in combination do not meet two of the basic criteria for establishing *prima facie* obviousness of claims 10 and 35. Neither of the references discloses applicants' limitations and there is no motivation for providing such limitations in the Maitra invention.

CLAIMS 12 AND 37

Claim 12 reads as follows:

12. The method of claim 10 wherein the software package causes the power utilized in executing the software package to be minimized after completion of the background tasks.

Apparatus claim 37 contains the same limitations as claim 12.

The only reference cited by the examiner for the obviousness of claims 12 and 37 is Maitra. The passage cited by the examiner (col. 2, lines 39-48) summarizes the Maitra invention as a method and apparatus for operating a microprocessor which reduces the power consumption and heat dissipation of the microprocessor. Maitra accomplishes this result by tailoring the clock speed of the microprocessor to the computing requirements of the application being executed (col. 2, lines 43-59). Maitra says nothing about the power utilized in executing the software package being minimized after completion of the background tasks. Maitra never discusses foreground and background tasks.

Since Maitra does not disclose the limitations of claims 12 and 37, there is obviously no motivation for a person skilled in the art for incorporating such limitations in the Maitra invention.

Thus, Maitra does not meet two of the basic criteria for establishing *prima facie* obviousness of claims 12 and 37. Maitra does not disclose applicants' limitations and there is no motivation for providing such limitations in the Maitra invention.

CLAIMS 11 AND 36

Claim 11 reads as follows:

11. The method of claim 10 wherein a background task is an infinite loop.

Apparatus claim 36 contains the same limitation as claim 11.

The examiner cites Davidson et al., col. 17, lines 32-58, as disclosing a background task which is an infinite loop.

The infinite loop of Davidson et al. is not a background task. It is a result of replacing a conditional branch instruction from a live code block to a dead code or rarely-executed code

block by a conditional branch-to-self instruction, thereby saving memory storage space. The occurrence of the infinite loop that results from the execution of the branch-to-self instruction is detected by a monitor process which causes the program to branch to the rarely executed code. Davidson et al., col. 17, lines 3-14. In other words, the occurrence of an infinite loop is an indication that the program should be redirected to executing the rarely executed code. The occurrence of an infinite loop as a means of redirecting the execution of a program is not a disclosure of applicants' background task infinite loop.

Thus, Maitra, Reznak, and Davidson et al. in combination do not meet two of the basic criteria for establishing *prima facie* obviousness of claims 11 and 36. The combination of references does not disclose applicants' claim limitations and there is no motivation for providing such limitations in the Maitra invention.

CLAIMS 13 AND 38

Claim 13 reads as follows:

13. The method of claim 1 wherein a failure in the execution of a software package causes information to be logged in a failure log.

Apparatus claim 38 contains the same limitation as claim 13.

The examiner cites Yen, col. 6, lines 49-59, as disclosing a failure in the execution of a software package causing information to be logged in a failure log.

The cited passage discloses a recovery system which only logs information relating to a startup failure. Failure in startup of a software package is not a disclosure of a failure to execute the software package.

Thus, Maitra and Yen in combination do not meet two of the basic criteria for establishing *prima facie* obviousness of claims 13 and 38. The combination of references does not disclose applicants' claim limitation and there is no motivation for providing such a limitation in the Maitra invention.

CLAIMS 14 AND 39

Claim 14 reads as follows:

14. The method of claim 13 wherein a failure in execution is linked to the software package that caused the failure.

Apparatus claim 39 contains the same limitation as claim 14.

The examiner cites Yen, col. 5, lines 1-16, and Figs. 3 and 9, as disclosing a failure in the execution of a software package being linked to the software package that caused the failure.

The Yen patent is for a system for recovering from certain types of system software startup problems whereby if an error is detected that would normally result in a startup failure, the computer's startup routine branches to an alternate startup application. Yen, Abstract. The passage cited by the examiner only discloses the options a user has available when a startup failure occurs. There is no disclosure pertaining to multi-tasking computer systems and failures in execution of one software package caused by another software package.

Maitra and Yen in combination do not meet two of the basic criteria for establishing *prima facie* obviousness of claims 14 and 39. The combination of references does not disclose applicants' claim limitation and there is no motivation for providing such a limitation in the Maitra invention.

CLAIMS 15 AND 40

Claim 15 reads as follows:

15. The method of claim 13 wherein quality of performance in executing a software package is represented by one or more performance-quality parameters, values of the one or more performance-quality parameters being determined from the information logged in a failure log, the execution of a software package being subject to a plurality of execution options, an execution option being selected on the basis of one or more performance-quality parameter values.

Apparatus claim 40 contains the same limitation as claim 15.

The examiner cites Harper et al. as disclosing the limitations of claim 15. The Harper et al. invention is a method and system for doing the following (col. 3, lines 1-59):

- learning how to predict outage of software system;
- predicting software outages;
- rejuvenating software; and
- predicting impending resource exhaustion and aging.

None of these tasks has anything to do with the limitations of claim 15. Nothing is disclosed concerning "quality of performance". Nothing is disclosed concerning "performance-quality parameters". Nothing is disclosed concerning the determination of the values of the "performance-quality parameters" from information logged in a failure log. Nothing is disclosed concerning the execution of a software package being subject to a plurality of "execution options". And nothing is disclosed concerning an "execution option" being selected on the basis of one or more "performance-quality parameter" values.

The examiner cites a 15-line passage from Harper et al. (col. 13, lines 35-50) as disclosing the entirety of the limitations of claim 15. Instead, the passage discusses the gathering of information for purposes of software rejuvenation and resource exhaustion—items which have nothing to do with applicants' claim 15.

Maitra and Harper et al. in combination do not meet two of the basic criteria for establishing *prima facie* obviousness of claims 15 and 40. The combination of references does not disclose applicants' claim limitations and there is no motivation for providing such limitations in the Maitra invention.

CLAIMS 16 AND 41

Claim 16 reads as follows:

16. The method of claim 15 wherein the plurality of execution options are user configurable.

Apparatus claim 41 contains the same limitation as claim 16.

Since Harper et al. does not disclose "execution options" (see discussion above under the **CLAIMS 15 AND 40**), a disclosure by Harper et al. that the "execution options" are user configurable is inconceivable. The examiner, however, cites Harper et al.'s Fig. 5 as being such a disclosure. Fig. 5 has to do with symptom-based software rejuvenation and has nothing to do with "execution options" that are user configurable.

Maitra and Harper et al. in combination do not meet two of the basic criteria for establishing *prima facie* obviousness of claims 16 and 41. The combination of references does not disclose applicants' claim limitation and there is no motivation for providing such a limitation in the Maitra invention.

CLAIMS 17 AND 42

Claim 17 reads as follows:

17. The method of claim 15 wherein performance-quality parameters include the number of failures and/or the rate of failures for one or more classes of failures recorded in a software package's failure log.

Apparatus claim 42 contains the same limitation as claim 17.

Since Harper et al. does not disclose "performance-quality parameters" (see discussion above under the **CLAIMS 15 AND 40**), an enumeration by Harper et al. of some of the "performance-quality parameters" seems unlikely. The examiner, however, cites col. 2, lines 32-35, of Harper et al. as being such a disclosure. However this passage simply states that in a co-pending application "it was described how to periodically rejuvenate all or part of a software system to reduce its failure rate to its initial, lower lever, based on time." This is not a disclosure of the claim-17 limitations. The examiner also cited Harper et al.'s Fig. 1 which shows a plot of software failure rate vs. time. This also is not a disclosure of the claim-17 limitations.

Maitra and Harper et al. in combination do not meet two of the basic criteria for establishing *prima facie* obviousness of claims 17 and 42. The combination of references does not disclose applicants' claim limitation and there is no motivation for providing such a limitation in the Maitra invention.

CLAIMS 23 AND 48

Claim 23 reads as follows:

23. The method of claim 1 wherein one or more of the plurality of software packages are independently compiled, linked, and loaded.

Apparatus claim 48 contains the same limitation as claim 23.

The examiner cites Potter et al. (col. 7, lines 40-57) as disclosing the limitation of claim 23. Potter et al. discloses a pattern recognition system 10 which recognizes patterns by means of the pattern recognition software package consisting of application 11 and data collectors and data manipulation portions 12. Potter et al., col. 7, lines 23-39. The passage cited by the examiner states that the pattern recognition software package is "written in source code which is compiled, linked and loaded by the Operating System 13, and which may incorporate subroutines included in the Run Time Library 15. This is not a disclosure of a method for repetitively executing a plurality of software packages (Claim 1) "wherein one or more of the plurality of software packages are independently compiled, linked, and loaded." The disclosure that an application consisting of two linked tasks is compiled, linked, and loaded by an operating system is not a disclosure of a multi-tasking computer system wherein on or more software packages are independently compiled, linked, and loaded.

Maitra and Potter et al. in combination do not meet two of the basic criteria for establishing *prima facie* obviousness of claims 23 and 48. The combination of references does not disclose applicants' claim limitation and there is no motivation for providing such a limitation in the Maitra invention.

CLAIMS 24 AND 49

Claim 24 reads as follows:

24. The method of claim 1 wherein a software package has its own stack, the software package's stack being selected prior to executing the software package.

Apparatus claim 49 contains the same limitation as claim 24.

The examiner cites Circello et al. as disclosing the limitations of claim 24. Circello et al. discloses "a data processor and method of operating a data processing system in which a single system stack pointer may be used to create records of both system and user stack operations when hardware support for alignment of such stack operands is optional." (Col. 3, lines 6-10) This is not a disclosure of a method for repetitively executing a plurality of software packages (Claim 1) "wherein a software package has its own stack, the software package's stack being selected prior to executing the software package."

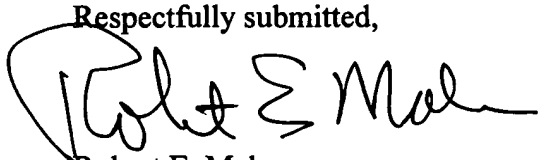
The examiner points specifically to Circello et al., col. 1, lines 35-51, which mentions that "user programs routinely manipulate the contents of the stack pointer which defines the top of the stack." However, the mere mention of "user programs" and "stack" in the same sentence does not disclose a multi-tasking computer structure wherein one or more of the software packages being executed may have their own stacks. Nor is it a disclosure of the software package's stack being selected prior to executing the software package.

Maitra and Circello et al. in combination do not meet two of the basic criteria for establishing *prima facie* obviousness of claims 24 and 49. The combination of references does not disclose applicants' claim limitation and there is no motivation for providing such a limitation in the Maitra invention.

* * * * *

The claims now appear to be in condition for allowance and such action is respectfully requested.

Respectfully submitted,



Date: 01/16/04
Telephone: (310) 459-8728

Robert E. Malm
Reg. No. 34,662